Probabilistic Robotics

Planning and Control:

Markov Decision Processes

Problem Classes

- Deterministic vs. stochastic actions
- Full vs. partial observability
- Today how to make decisions under uncertainty

Uncertainty and decisions

- Previously how to do state estimation under uncertainty
- Uncertainty can affect how the robot makes decisions
- How to encode preferences, between different outcomes of the planes (e.g. going to the airport – lots of options, risks)
- Utility theory – reasoning about preferences (utility – quality of being useful)
- Every state has some utility

- Decision theory = probability theory + utility theory

- Principle of maximum expected utility – agent is rational if it chooses an action with the highest expected utility

Designing control systems

- Often in addition to stability, observability, controllability, we want to have some optimality
- Such that the goal it that the trajectory will maximize certain performance index (e.g. time travelled, fuel cost, quadratic cost for trajectory tracking …)
- Using techniques from calculus of variations to solve for functions which maximize the performance index V

- Special class of systems n-stage decision processes
- Find such V and choices of action such that the V is maximal
- Blackboard example: Recursive computation of V in deterministic case (in case of grid world similar to waverfront planner)
- Principle of dynamic programming – decompose the problem in n-stages; at each stage relaxation

- Deterministic case: find such sequence of actions that the performance is maximized
- Consider simple performance index – sum of individual rewards x – state, u- control, U – utility

$$U(x_0,\cdots,x_n,u_1,\cdots,u_n) = R(x_0)+\cdots+R(x_n)$$

- Idea recursive computation of U for each state
- $U_n(x_1) = \max_u[R(u_1,x_1)+U_{n-1}(x_2)]$ or
- $U_n(x) = \max_u[R(u,x)+U_{n-1}(f(s,x))]$, example:

$R(x)=0$ if s is goal $R(x)=-1$ otherwise

$U(x)$ Desirability of a state

-1	-1	-1	0
-1		-1	-1
-1	-1	-1	-1

-3	-2	-1	0
-4		-2	-1
-5	-4	-3	-2

Optimal policy

- Once we have the optimal value function
- Policy: choose at each instance a state which with maximal utility

$$\pi = \max_u V(f(x,u))$$

$$\pi : X \to U$$

-3	-2	-1	0
-4		-2	-1
-5	-4	-3	-2

->	->	->	0
Down/up		->	up
->	->	->	up

- Next what if the outcomes of actions are uncertain

Markov decision processes

- Framework for represention complex multi-stage decision problems in the presence of uncertainty
- Efficient solutions
- Models the dynamics of the environment under different actions
- Outcomes of actions are uncertain – probabilistic model
- Markov assumptions : next state depends in the previous state, and action not the past

Markov Decision Process

- Formal definition
- 4-tuple (X, U, T, R)
- Set of states X - finite
- Set of actions A – finite
- Transition model $T : X \times U \times X \to [0,1]$

  Transition probability for each action, state
- Reward model

$$X \times U \times X \to R$$

- Utility of a state under given policy – expected sum of discounted rewards

$$U^{\pi}(x) = E[\sum_{t=0}^{\infty} \gamma^{t} R_{t}(x_{t}) \mid \pi]$$

- Goal: find such policies which maximize sum of expected rewards

Types of rewards

- Reward structure: additive rewards

$$U(x_0, x_1, \ldots, x_n): \quad R(x_0) + R(x_1) + \ldots + R(x_n)$$

- Discounted rewards

$$U(x_0, x_1, \ldots, x_n): \quad R(x_0) + \gamma R(x_1) + \ldots + \gamma^n R(x_n)$$

- Preference for current rewards over future rewards (good model for human and animal preferences over time)
- How to deal with the infinite rewards ? Make sure that the utility of the infinite sequence is finite
- Design proper policies which are guaranteed to reach the final state
- Compare policies based on average reward per step

Utility of the state

- How good the state is – defined in terms of sequence
- Utility of the state is expected utility of sequences which may follow that state

$$U^{\pi}(x) = E[\sum_{t=0}^{\infty} \gamma^t R(x_t) \mid \pi, x_0 = x]$$

- Distinction between reward and utility
- Goal: Find the best policy

$$\pi^*: X \rightarrow U$$

Optimal Payoff

- Bellman equation: set of linear constraints, given a policy
- We can compute the utility of each state (value function) under policy

$$U^{\pi}(x) = R(x) + \gamma \sum_{s'} T(x,u,x')U(x')$$

- One equation per state, n states n equations, solve for U
- Find such policy which maximizes the payoff

$$U^{*}(x) = \max_{\pi} U^{\pi}(x)$$

- We know how to compute values function (solve linear eq.)
- How to compute optimal policy – there are exponentially many sequences of actions

Value Iteration

- Calculation of optimal policies
- Calculate utility of each state and use state utilities to select the next action
- Given utility of a state

$$U^{\pi}(s) = E(\sum_{t=0}^{\infty} \gamma^t R(x_t))$$

$$U^{\pi}(x) = E[R(x) + \gamma(R(x_1) + \gamma R(x_2) + \cdots)]$$

- Reward in current state + value function for next state
- Bellman equation

$$U^{\pi}(x) = R(s) + \gamma \max_u \sum_{s'} T(x,u,x')U(x')$$

- Example

Value Iteration

- Bellman equation

$$U_n^{\pi}(x) = R(x) + \gamma \max_u \sum_{s'} T(x,u,x')U_{n-1}(x')$$

- Recursive computation
- Iterate while

$$(U_n^{\pi}(x) - U_{n-1}(x)) > \varepsilon$$

- If the consecutive iterations differ little, fix point is reached
- Value iteration converges

Value iteration

- Compute the optimal value function first, then the policy
- N states – N Bellman equations, start with initial values, iteratively update until you reach equilibrium
- 1. Initialize V; For each state x

$$U_n(x) = R(x) + \gamma \max_a \sum_{x'} T(x,u,x') U_{n-1}(x')$$

- If $\left| U_n(x) - U_{n-1}(x) \right| > \delta$ then $\delta \leftarrow \left| U_n(x) - U_{n-1}(x) \right|$
- until

$$\delta < \varepsilon(1-\gamma)/\gamma$$

- Return U
- Optimal policy can be obtained before convergence of value iteration

Example

- Adopted from Russell and Norvig AI
- Robot navigating on the grid
- 4 actions – up, down, left, right
- Effects of moves are stochastic, we may end up in other state then indented with non-zero probability
- Reward +1 for reaching the goal, -1 close to ditch, -0.04 for other states
- Goal: find the policy sequence of actions $\pi : \; x_t \rightarrow \; u_t$
- First compute the utility of each state using value iteration

0.81	0.86	0.91	+1
0.76		0.66	-1
0.70	0.66	0.61	0.38

Utility of the states

Transition model:

T(x, u, x')

Up = 0.8 up 0.1 left 0.1 right

Left = ...

Right = ...

Down = ...

Example

- Robot navigating on the grid - up, down, left, right
- Reward +1 for reaching the goal, -1 for going to (4,2)
- R(s) = -0.04 small negative reward for visiting non-goal states (penalize wandering around0
- Goal: find the policy sequence of actions
- Solution

$$\pi: \quad x_t \rightarrow u_t$$

0.81	0.86	0.91	+1
0.76		0.66	-1
0.70	0.66	0.61	0.38

→	→	→	+1
↑		↑	-1
↑	←	←	←

- Idea: calculate utilities of a state, select optimal action in each state – one that maximizes utility

Example

- 4 actions – up, down, left, right
- Reward +1 for reaching the goal, -1 close to ditch, -0.04 for other states

$$U(1,1) = -0.04 + \gamma \max($$
$$0.8U(1,2) + 0.1U(2,1) + 0.1U(1,1),$$ best action is up
$$0.9U(1,1) + 0.1U(1,2),$$
$$0.9U(1,1) + 0.1U(2,1),$$
$$0.8U(2,1) + 0.1U(1,2) + 0.1U(1,1)]$$

3	0.81	0.86	0.91	+1
2	0.76		0.66	-1
1	0.70	0.66	0.61	0.38
	1	2	3	4

Utility of the states

$$\pi : \quad x_t \longrightarrow u_t$$

Transition model:

T(x, u, x')

Up = 0.8 up 0.1 left 0.1 right

Left = …

Right = …

Down = …

Policy Iteration

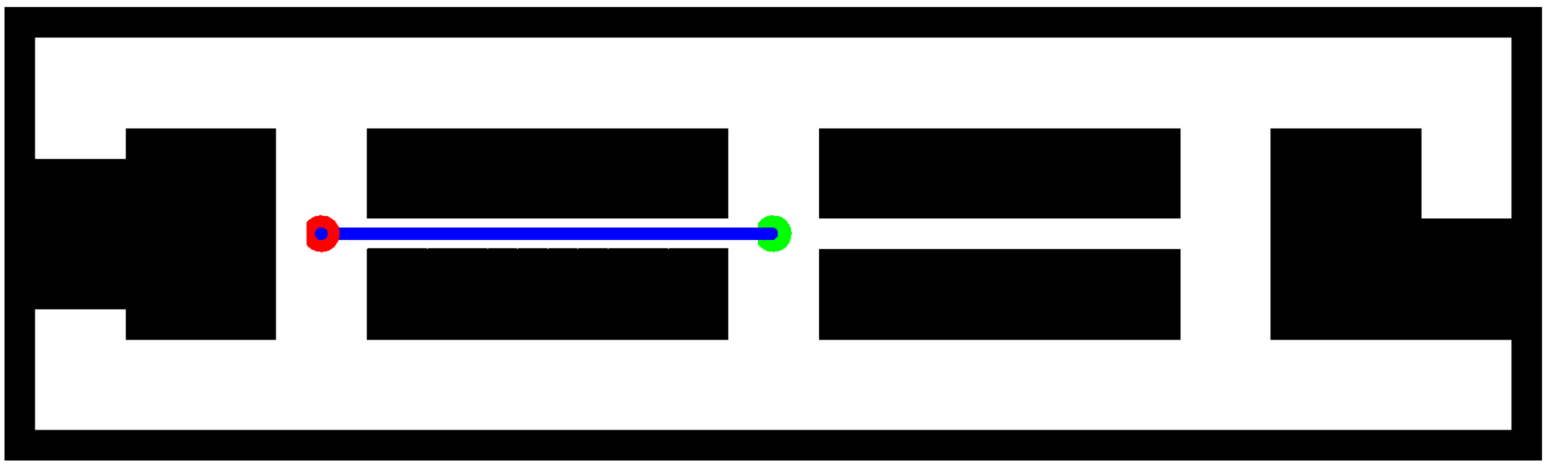
- Takes policy and computes its value
- Iteratively improved policy, until it cannot be further improved
- 1. Policy evaluation – calculate the utility of each state under particular policy π_i
- 2. Policy improvement – Calculate new MEU policy, using one-step look-ahead based on π_{i+1}
- 1. Initialize policy
- 2. Evaluate policy get V; For each state do if

$$\max_u \sum_{x'} T(x,u,x')U(x') > \sum_{s'} T(x,\pi(x),x')U(x')$$

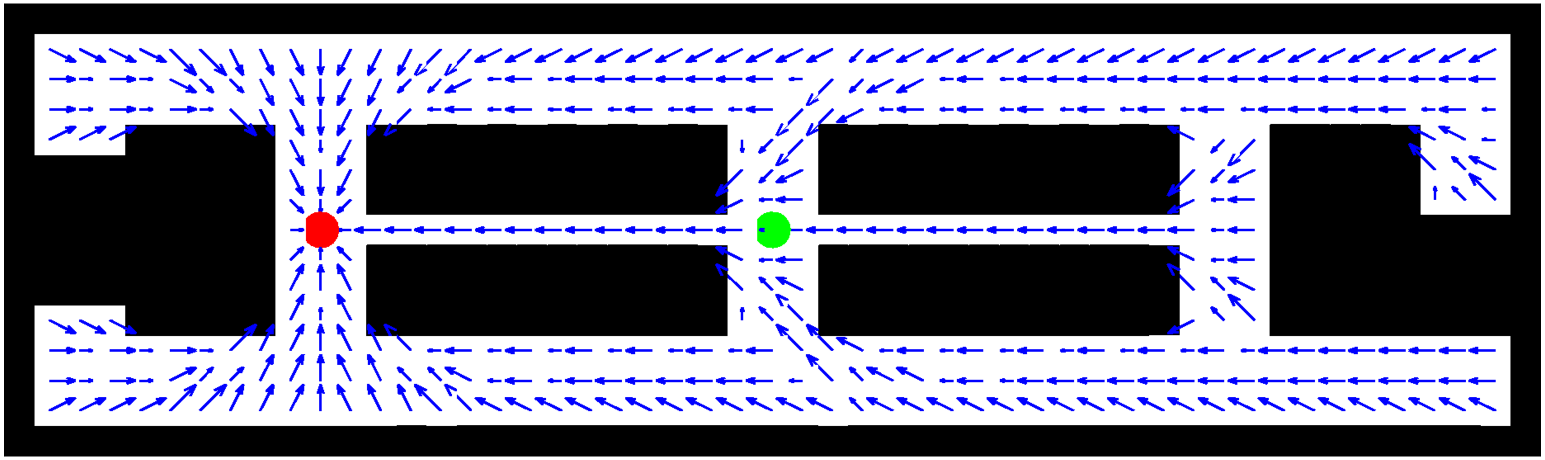
- Until unchanged

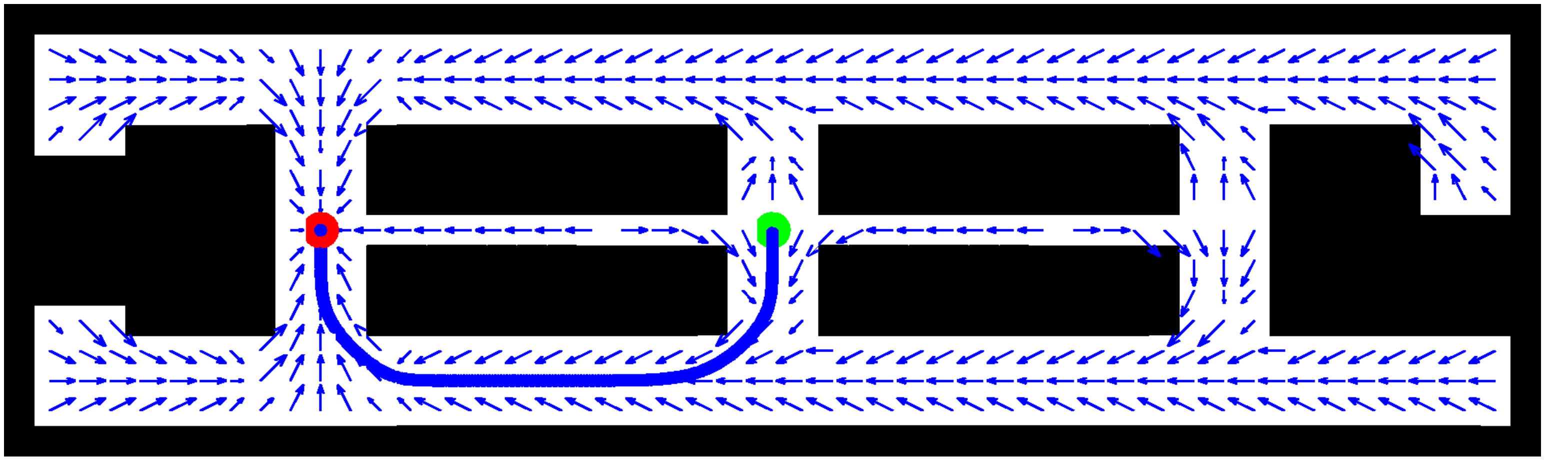
$$\pi(s) \leftarrow \arg\max_u \sum_{x'} T(x,u,x')U(x')$$

Deterministic, fully observable



Stochastic, Fully Observable





Stochastic, Partially Observable